A. Appendix

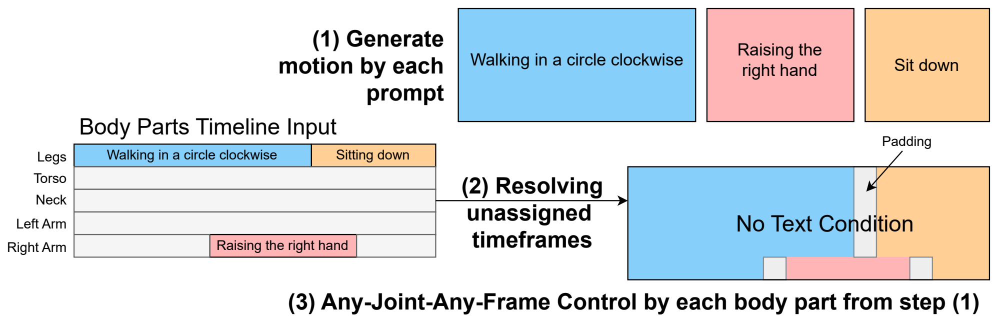
A.1. Overview

The supplementary material is organized into the following sections:

Video visualization can be found at `https://www.ekkasit.com/ControlMM-page/`

A.2. Comparision to STMC

The STMC [38] setting involves different controllable joints and longer motion sequences, making direct comparison infeasible without re-training the *Logits Regularizer*. Fortunately, *Logits Optimization* enables zero-shot objective control, allowing us to achieve the STMC setting without retraining. We adopt the following steps: **(1) Generate motions for each prompt**: Motions are generated separately without joint control. **(2) Resolving unassigned timeframes**: Following STMC, we apply the SINC heuristic to fill in unassigned joints based on the sparse body-part timeline input. **(3) Any-Joint-Any-Frame Control by each body part**: We use the control conditions from steps (1) and (2), and remove joint control from connected timeframes to create padding, enabling smooth transitions between adjacent motion segments. Finally, the full motion is generated using these joint control signals without any text prompt, effectively generating body-part timelines onto a blank canvas.



(1) Generate motion by each prompt
Walking in a circle clockwise
Raising the right hand
Sit down
Body Parts Timeline Input
Legs
Torso
Neck
Left Arm
Right Arm
Walking in a circle clockwise
Sitting down
Raising the right hand
Padding
(2) Resolving unassigned timeframes
No Text Condition
(3) Any-Joint-Any-Frame Control by each body part from step (1)

Figure 5. Generating body parts timeline for STMC setting.

We evaluate two variants: **'No Control Pad'**, where no control is applied in the padding regions, and **'Merge Control Pad'**, where the optimization weight gradually decreases from adjacent body-part controls. Our method outperforms STMC across all metrics, as shown in Tab. 6. Note that performance could be further improved by re-training the *Logits Regularizer* for the STMC setting.

Method	Per-crop semantic correctness				Realism	
	R@1 ↑	R@3 ↑	TMR-Score ↑ M2T	M2M	FID ↓	Transition distance ↓
Ground truth	55.0	73.3	0.748	1.000	0.000	1.5
STMC-MotionDiffuse	24.8	46.7	0.660	0.632	0.531	**1.5**
STMC-MDM	25.1	46.0	0.641	0.633	0.606	2.4
Our (No Control Pad)	**38.3**	**58.1**	**0.688**	**0.654**	**0.511**	1.6
Our (Merge Control Pad)	**34.8**	**55.6**	**0.675**	**0.653**	**0.508**	**1.5**

Table 6. Quantitative comparison with STMC

A.3. Pseudo Code of MaskControl Inference

Algorithm 1 MaskControl Inference

```
Require: Masked Motion Model ($MMM$), Logits Regularizer ($LR$), mask scheduling function $\gamma(\cdot)$, spatial control signals $s$ (if any), text prompts $W$ (if any).
1: $X_{\overline{\mathbf{M}}} \leftarrow$ [Mask]                                   ▷ Start with all mask tokens
2: for all $t$ from 1 to $T$ do                                                  ▷ Unmask process in $T$ steps
3:     $\{\mathbf{f}\} \leftarrow LR(X_{\overline{\mathbf{M}}}, W, s; \phi)$                  ▷ Logits Regularizer
4:     $l \leftarrow MMM(X_{\overline{\mathbf{M}}}, \mathbf{p}, \{\mathbf{f}\}; \theta)$      ▷ Masked Motion Model
5:     for all $i$ from 1 to $I_l$ do                                            ▷ Logits Optimization
6:         $l_{i+1} = l_i - \eta \nabla_{l_i} L_s(l_i, s)$
7:     end for
8:     $X_{\overline{\mathbf{M}}} \leftarrow \gamma(l, t)$                    ▷ mask out tokens based on logits $l$ at time step $t$
9: end for
10: for all $i$ from 1 to $I_e$ do                                               ▷ Logits Optimization
11:     $e_c^{i+1} = e_c^i - \eta \nabla_{e_c^i} L_s(e_c^i, s)$
12: end for
13: return $Decoder(e_c)$
```

A.4. Implementation Details

We modified the MoMask [16] model by retraining it with a cross-entropy loss applied to all tokens, instead of just the masked positions. This retrained model serves as our pretrained base model, and we kept the default hyperparameter settings unchanged. To improve robustness to text variation, we randomly drop 10% of the text conditioning, which also allows the model to be used for Classifier-Free Guidance (CFG). The weight for Eq. 5 is set to $\alpha = 0.1$. We use a codebook of size 512, with embeddings of size 512 and 6 residual layers. The Transformer embedding size is set to 384, with 6 attention heads, each with an embedding size of 64, distributed across 8 layers. This configuration demonstrates the feasibility of converting between two different embedding sizes and spaces using the Differentiable Expectation Sampling (DES). The encoder and decoder downsample the motion sequence length by a factor of 4 when mapping to token space. The learning rate follows a linear warm-up schedule, reaching 2e-4 after 2000 iterations, using AdamW optimization. The mini-batch size is set to 512 for training RVQ-VAE and 64 for training the Transformers. During inference, the CFG scale is set to $cfg = 4$ for the base layer and $cfg = 5$ for the 6 layers of residual, with 10 steps for generation. We use pretrained CLIP model [41] to generate text embeddings, which have a size of 512. These embeddings are then projected down to a size of 384 to match the token size used by the Transformer. *Logits Regularizer* is a trainable copy of Masked Transformer with the zero linear layer connect to the output each layer of the Masked Transformer. During inference, *Logits Optimization* applies L2 loss with a learning rate of 0.06 for 100 iterations in *Logits Optimization* for each of the 10 generation steps and 600 iterations in the last unmasking step. We apply temperature of 1 for all 10 steps and 1e-8 for residual layers. We follow the implementation from [21, 51, 55], applying the spatial control signal only to joint positions and omitting rotations.

A.5. Full Evaluation on All Joint

Following the evaluation from OmniControl [55], Table 7 extends Table 2 by showing the evaluation for each joint individually. Our MaskControl outperforms SOTA across all metrics. 'Cross' is the random combination of joints can be found in

Sec. A.13

Table 7. Comparison of text-condition motion generation with spacial control signal on the HumanML3D. The first section, "Train on Pelvis Only," evaluates our model that was trained solely on the pelvis. The last section, "Train on All Joints", is trained on all joints and assessing performance for each one. The cross-section reports performance across various combinations of joints.

Method	**Joint**	**R-Precision Top-3 ↑**	**FID ↓**	**Diversity →**	**Foot Skating Ratio ↓**	**Traj. Err. (50 cm) ↓**	**Loc. Err. (50 cm) ↓**	**Avg. Err. ↓**
Real	-	0.797	0.002	9.503	-	0.0000	0.0000	0.0000
Train on Pelvis Only								
MDM		0.602	0.698	9.197	0.1019	0.4022	0.3076	0.5959
PriorMDM		0.583	0.475	9.156	0.0897	0.3457	0.2132	0.4417
GMD		0.665	0.576	9.206	0.1009	0.0931	0.0321	0.1439
OmniControl (on pelvis)	Pelvis	0.687	0.218	9.422	0.0547	0.0387	0.0096	0.0338
TLControl		0.779	0.271	9.569	-	0.0000	0.0000	0.0108
MotionLCM		0.752	0.531	9.253	-	0.1887	0.0769	0.1897
MaskControl (on pelvis)		**0.809**	**0.061**	**9.496**	**0.0547**	**0.0000**	**0.0000**	**0.0098**
Train on All Joints								
OmniControl		0.691	0.322	**9.545**	0.0571	0.0404	0.0085	0.0367
TLControl		0.779	0.271	9.569	-	0.0000	0.0000	**0.0108**
MaskControl		**0.804**	**0.071**	9.453	**0.0546**	**0.0000**	**0.0000**	0.0127
OmniControl		0.696	0.280	**9.553**	0.0692	0.0594	0.0094	0.0314
TLControl	Left Foot	0.768	0.368	9.774	-	0.0000	0.0000	0.0114
MaskControl		**0.804**	**0.076**	9.389	**0.0559**	**0.0000**	**0.0000**	**0.0072**
OmniControl		0.701	0.319	**9.481**	0.0668	0.0666	0.0120	0.0334
TLControl	Right Foot	0.775	0.361	9.778	-	0.0000	0.0000	0.0116
MaskControl		**0.805**	**0.074**	9.400	**0.0549**	**0.0000**	**0.0000**	**0.0068**
OmniControl		0.696	0.335	**9.480**	0.0556	0.0422	0.0079	0.0349
TLControl	Head	0.778	0.279	9.606	-	0.0000	0.0000	0.0110
MaskControl		**0.805**	**0.085**	9.415	**0.0538**	**0.0000**	**0.0000**	**0.0071**
OmniControl		0.680	0.304	**9.436**	0.0562	0.0801	0.0134	0.0529
TLControl	Left Wrist	0.789	0.135	9.757	-	0.0000	0.0000	0.0108
MaskControl		**0.807**	**0.093**	9.374	**0.0541**	**0.0000**	**0.0000**	**0.0051**
OmniControl		0.692	0.299	**9.519**	0.0601	0.0813	0.0127	0.0519
TLControl	Right Wrist	0.787	0.137	9.734	-	0.0000	0.0000	0.0109
MaskControl		**0.805**	**0.099**	9.340	**0.0539**	**0.0000**	**0.0000**	**0.0050**
OmniControl	Average	0.693	0.310	**9.502**	0.0608	0.0617	0.0107	0.0404
MaskControl		**0.805**	**0.083**	9.395	**0.0545**	**0.0000**	**0.0000**	**0.0072**
OmniControl	Cross	0.672	0.624	9.016	0.0874	0.2147	0.0265	0.0766
MaskControl		**0.811**	**0.049**	**9.533**	**0.0545**	**0.0000**	**0.0000**	**0.0126**

A.6. Inference speed, quality, and errors

We compare the speed of three different configurations of our model against state-of-the-art methods as shown in Table 8. The first setting, **MaskControl-Fast**, uses only 100 iterations of *Logits Optimization* in the last step of unmasking process. This setup achieves results comparable to OmniControl, but is over 20 times faster. It also slightly improves the Trajectory and Location Errors, while the FID score is only 25% of OmniControl's, indicating high generation quality. The second setting, **MaskControl-Medium**, increases the *Logits Optimization* to 600 iterations, which further improves accuracy. The Location Error is reduced to zero, although the FID score slightly worsens. Lastly, the **MaskControl-Accurate** model, which is the default setting used in other tables in this paper, uses 600 iterations of *Logits Optimization* in the last step of unmasking process and 100 iterations of 1-9 step *Logits Optimization* in the last step of unmasking process. This configuration achieves extremely high accuracy, with both the Trajectory and Location Errors reduced to zero and the Average Error below 1 cm (0.98 cm). Importantly, these settings can be adjusted during inference without retraining the model, making them suitable for both real-time and high-performance applications. Fig. 8 compares FID, Location Error, and speed.

Table 8. Comparison of Motion Generation Performance with Speed and Quality Metrics

Model	Speed ↓	R-Precision Top-3 ↑	FID ↓	Diversity →	Foot Skating Ratio ↓	Traj. Err. (50 cm) ↓	Loc. Err. (50 cm) ↓	Avg. Err. ↓
MDM	10.14 s	0.602	0.698	9.197	0.1019	0.4022	0.3076	0.5959
PriorMDM	18.11 s	0.583	0.475	9.156	0.0897	0.3457	0.2132	0.4417
GMD	132.49 s	0.665	0.576	9.206	0.1009	0.0931	0.0321	0.1439
OmniControl	87.33 s	0.687	0.218	9.422	0.0547	0.0387	0.0096	0.0338
MaskControl-Fast	4.94 s	0.808	0.059	9.444	0.0570	0.0200	0.0075	0.0550
MaskControl-Medium	25.23 s	0.806	0.069	9.425	0.0568	0.0005	0.0000	0.0124
MaskControl-Accurate	71.72 s	0.809	0.061	9.496	0.0547	0.0000	0.0000	0.0098

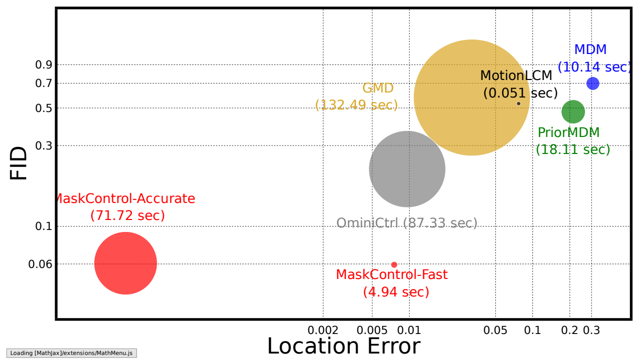


Figure 6. Comparison of FID score, spatial control error, and motion generation speed (circle size) for our accurate and fast models comparing to state-of-the-art models. The closer the point is to the origin and the smaller the circle, the better performance.

A.7. Speed of each component

We report the inference time for each component in Table 9, with all measurements taken on an NVIDIA A100. The **Base** model, which includes only the Masked Transformer with Residual layers and Decoder (without any spatial control signal module), has an inference time of 0.35 second. The **Logits Regularizer** is highly efficient, requiring only 0.24 seconds for inference. The **Logits Optimization** takes 24.65 seconds for unmasking step 1-9 and 46.5 seconds in the last step, respectively. In total, the **MaskControl-Accurate** model has a generation time of 71.73 seconds. Note that this setting is using 100 iterations of **Logits Optimization** for 1-9 steps and 600 iterations of the last step.

Table 9. Inference time of each component

	Base	Logits Regularizer	Logits Optimization (1-9 steps)	Logits Optimization (last step)	Full
Speed in Seconds	0.35	0.24	24.65	46.5	71.73

A.8. Quantitative result for all joints of MaskControl-Fast

Table 10 presents the evaluation results for MaskControl-Fast, which uses 100 iterations of **Logits Optimization**. This evaluation includes a “cross” assessment that evaluates combinations of different joints, as detailed in Section A.13. The results can be compared to those of the full model (MaskControl-Accurate) and state-of-the-art models shown in Table 2. Additionally, ”lower body” refers to the conditions involving the left foot, right foot, and pelvis, which allows for the evaluation of upper body editing tasks, as illustrated in Table 3.

A.9. Ablation on less number of generation step

In this section, we perform an ablation study on the number of steps used in the generation process. Following the MoMask architecture [16], we adopt the same setting of 10 steps for generation. However, the integration of *Logits Optimization* and the *Logits Regularizer* enhances the quality of the generated outputs with fewer steps, as demonstrated in Table 11. Notably,

Table 10. Quantitative result for all joints of MaskControl-Fast

Joint	**R-Precision Top-3** ↑	**FID** ↓	**Diversity** ↑	**Foot Skating Ratio** ↓	**Traj. Err. (50 cm)** ↓	**Loc. Err. (50 cm)** ↓	**Avg. Err.** ↓
pelvis	0.806	0.067	9.453	0.0552	0.0446	0.0151	0.0691
left foot	0.806	0.074	9.450	0.0561	0.0495	0.0105	0.0484
right foot	0.808	0.069	9.416	0.0566	0.0453	0.0099	0.0469
head	0.810	0.080	9.411	0.0555	0.0525	0.0148	0.0665
left wrist	0.809	0.085	9.380	0.0545	0.0467	0.0108	0.0534
right wrist	0.807	0.095	9.387	0.0549	0.0498	0.0113	0.0538
Average	0.808	0.079	9.416	0.0555	0.0481	0.0121	0.0563
cross	0.812	0.050	9.515	0.0545	0.0330	0.0101	0.0739
lower body	0.807	0.084	9.396	0.0491	0.0312	0.0050	0.0633

with just 1 step, the results are already comparable to those achieved by TLControl [51]. Furthermore, after 4 steps, the evaluation metrics are on par with those obtained after 10 steps.

Table 11. Quantitative result for different number of steps

# of steps	**R-Precision Top-3** ↑	**FID** ↓	**Diversity** →	**Foot Skating Ratio** ↓	**Traj. Err. (50 cm)** ↓	**Loc. Err. (50 cm)** ↓	**Avg. Err.** ↓
1	0.779	0.276	9.353	0.0545	0.0002	0.0000	0.0110
2	0.792	0.118	9.436	0.0530	0.0001	0.0000	0.0100
4	0.806	0.068	9.468	0.0543	0.0001	0.0000	0.0098
6	0.809	0.063	9.478	0.0545	0.0001	0.0000	0.0098
8	0.810	0.059	9.511	0.0543	0.0001	0.0000	0.0098
10	0.809	0.061	9.496	0.0547	0.0000	0.0000	0.0098

To further investigate the influence of *Logits Optimization* and the *Logits Regularizer* for lesser steps, we remove these components and experiment with various numbers of steps and apply *Logits Regularizer* only last step, as shown in Table 12. Reducing the number of steps significantly decreases the quality of the generated outputs, resulting in an FID score of 1.196 with only 1 step. Even with 10 steps, the FID score remains at 0.190, highlighting the improvements by integrating *Logits Optimization* and the *Logits Regularizer*.

Table 12. Quantitative result for different number of steps without *LogitsOptimization* and *Logits Regularizer*

# of steps	**R-Precision Top-3** ↑	**FID** ↓	**Diversity** →	**Foot Skating Ratio** ↓	**Traj. Err. (50 cm)** ↓	**Loc. Err. (50 cm)** ↓	**Avg. Err.** ↓
1	0.716	1.196	8.831	0.0715	0.0070	0.0006	0.0271
2	0.758	0.462	9.182	0.0672	0.0067	0.0005	0.0276
4	0.782	0.238	9.236	0.0628	0.0066	0.0005	0.0281
6	0.787	0.203	9.276	0.0614	0.0061	0.0005	0.0282
8	0.787	0.193	9.272	0.0613	0.0062	0.0005	0.0283
10	0.786	0.190	9.294	0.0616	0.0063	0.0005	0.0283

A.10. Analysis of *Logits Optimization* and *Logits Regularizer*

To understand the impact of *Logits Optimization* and *Logits Regularizer* on the generation process, we visualize the maximum probability for each token prediction from the Masked Transformer. The model predicts 49 tokens over 10 steps. We show results both before and after applying *Logits Optimization*, and with and without the *Logits Regularizer*. The maximum probability can be expressed as the relative value of the logits corresponding to all codes in the codebook in the specific token position and step, as computed by the Softmax function. We visualize the output using the Softmax function instead of Gumbel-Softmax. By removing the Gumbel noise, Gumbel-Softmax reduces to a regular Softmax function:

$$p_i = \frac{\exp(\ell_i)}{\sum_{j=1}^{k} \exp(\ell_j)}$$

The generation is conditioned by the text prompt, "a person walks in a circle counter-clockwise" with control over the pelvis and right hand throughout the entire trajectory. In the plot, darker blue colors represent lower probabilities (0), while yellow represents higher probabilities (1).

Without *Logits Regularizer*

In the first step (step 0), the probability is low but increases significantly in the subsequent steps. After applying *Logits Optimization*, the probability improves slightly, as shown in Fig. 7 and 8. Eventually, the probability saturates in the later steps (see Figure 11). Since the probability of most token predictions approaches one, *Logits Optimization* cannot further modify the logits, preventing any updates to the trajectory.

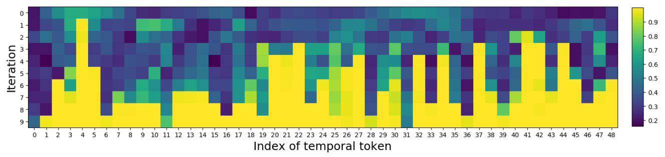


Figure 7. The maximum probability of the each token **without** *Logits Regularizer* **before** *Logits Optimization* of each all 49 tokens and 10 steps.

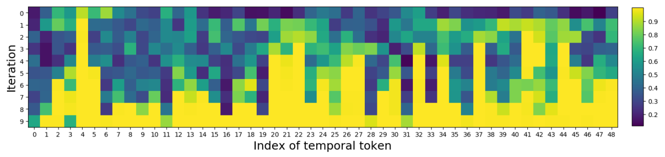


Figure 8. The maximum probability of the each token **without** *Logits Regularizer* **after** *Logits Optimization* of each all 49 tokens and 10 steps.

With *Logits Regularizer*

With the introduction of the *Logits Regularizer*, the probability of token predictions is significantly higher in the initial step compared to the scenario without the *Logits Regularizer*, as illustrated in Figures 9 and 10. Moreover, the maximum probability does not saturate to one, indicating that there is still room to adjust the logits for trajectory editing.

This enhancement leads to improved generation quality within fewer steps, as detailed in Section A.9. Notably, just 4 steps using the *Logits Regularizer* yield a quality comparable to that achieved in 10 steps without it, where the latter still exhibits suboptimal quality and high average error.

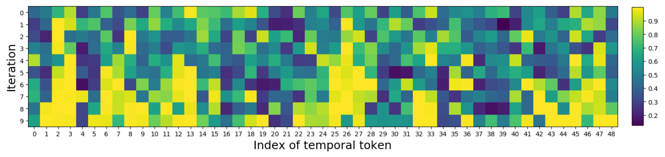


Figure 9. The maximum probability of the each token **with** *Logits Regularizer* **before** *Logits Optimization* of each all 49 tokens and 10 steps.

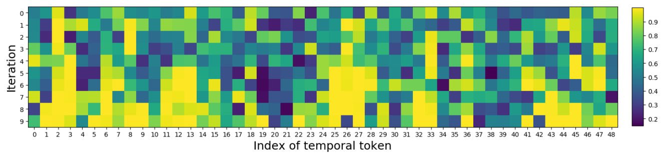


Figure 10. The maximum probability of the each token **with** *Logits Regularizer* **after** *Logits Optimization* of each all 49 tokens and 10 steps.

Average of maximum probability of all tokens in each step To clearly illustrate the increasing probability or confidence of the model predictions across all 10 steps, as shown in Fig. 11. In this figure, the blue line represents the average probability of token predictions **With the *Logits Regularizer***, while the red line denotes the average probability **Without the *Logits Regularizer***. The solid line indicates the average probability prior to the application of *Logits Optimization*. This shows that the probability increases significantly in the very first step for the **With the *Logits Regularizer***.

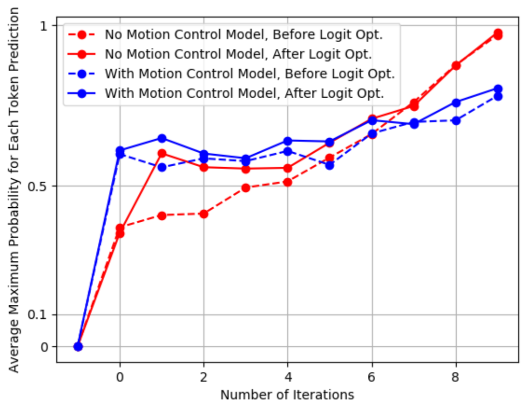


Figure 11. Average Maximum Probability for Each Token Prediction

A.11. Ambiguity of Motion Control Signal

In the image domain, pixel control signals can be directly applied, and uncontrolled regions are simply zeroed out. However, for motion control, zero-valued 3D joint coordinates are ambiguous: they may indicate that a joint is controlled with its target position at the origin in Euclidean space, or that the joint is uncontrolled. To resolve this ambiguity, we concatenate the joint control signal with the relative difference between the control signal and the generated motion, forming the final joint control guidance s.

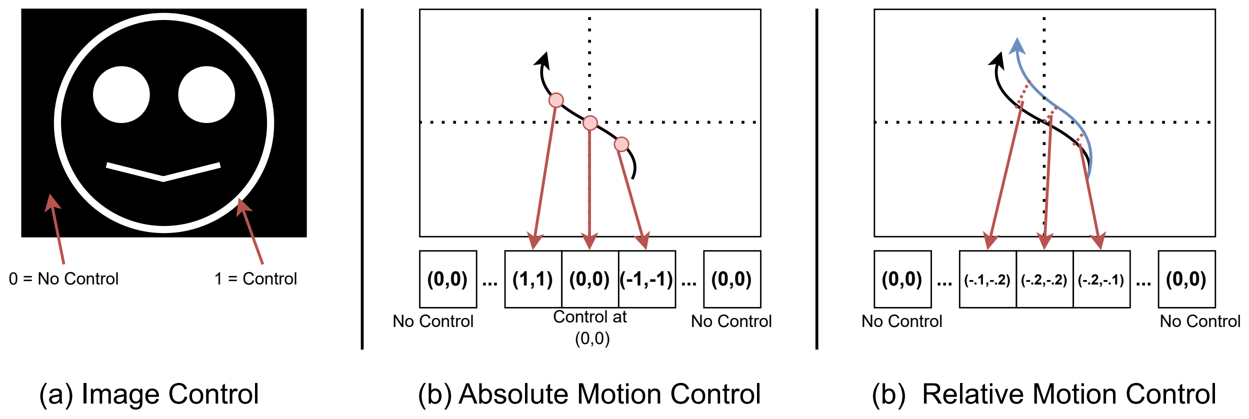


Figure 12. The difference between control signals: **(a) Image Control**: 0 means no control, 1 means control. **(b) Absolute Motion Control**: ambiguous between control signal at origin and no control. **(c) Relative Motion Control**: no ambiguity. Black curve: spatial control signal. Blue curve: decoded spatial signal from generated motion

A.12. Body Part Timeline Control

Generating multiple body parts based on their respective text prompts is not straightforward, as the HumanML3D dataset provides only a single prompt for each motion without specific descriptions for individual body parts. However, our model can conditionally generate outputs based on spatial signals, which allows us to manipulate and control the generation process. To achieve this, we first generate the entire body and motion for all frames. Next, we generate a new prompt related to the next body part, using the previously generated body parts as a condition. This process can be repeated multiple times to create motion for each body part based on its corresponding text prompt, as illustrated in Fig. 13.
It is important to note that this approach may lead to out-of-distribution generation since the model has not been trained on combinations of multiple body parts with their associated text prompts. However, our model handles out-of-distribution generation effectively due to the use of *Logits Optimization*.

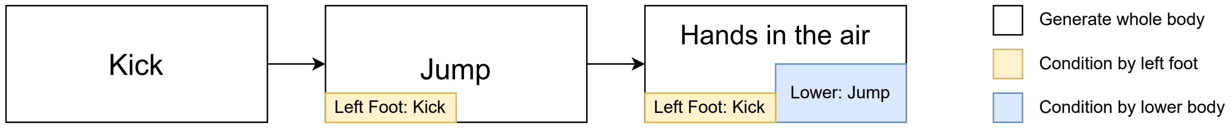


Figure 13. Process of generating body parts with multiple text inputs over a specific timeline

A.13. Cross Combination

We follow the evaluation *Cross Combination* from OmniControl [55], evaluating multiple combinations of joints as outlined in Table 2. A total of 63 combinations are randomly sampled during the evaluation process as follow.

1. pelvis
2. left foot
3. right foot
4. head
5. left wrist
6. right wrist
7. pelvis, left foot
8. pelvis, right foot
9. pelvis, head
10. pelvis, left wrist
11. pelvis, right wrist
12. left foot, right foot
13. left foot, head

14. left foot, left wrist
15. left foot, right wrist
16. right foot, head
17. right foot, left wrist
18. right foot, right wrist
19. head, left wrist
20. head, right wrist
21. left wrist, right wrist
22. pelvis, left foot, right foot
23. pelvis, left foot, head
24. pelvis, left foot, left wrist
25. pelvis, left foot, right wrist
26. pelvis, right foot, head
27. pelvis, right foot, left wrist
28. pelvis, right foot, right wrist
29. pelvis, head, left wrist
30. pelvis, head, right wrist
31. pelvis, left wrist, right wrist
32. left foot, right foot, head
33. left foot, right foot, left wrist
34. left foot, right foot, right wrist
35. left foot, head, left wrist
36. left foot, head, right wrist
37. left foot, left wrist, right wrist
38. right foot, head, left wrist
39. right foot, head, right wrist
40. right foot, left wrist, right wrist
41. head, left wrist, right wrist
42. pelvis, left foot, right foot, head
43. pelvis, left foot, right foot, left wrist
44. pelvis, left foot, right foot, right wrist
45. pelvis, left foot, head, left wrist
46. pelvis, left foot, head, right wrist
47. pelvis, left foot, left wrist, right wrist
48. pelvis, right foot, head, left wrist
49. pelvis, right foot, head, right wrist
50. pelvis, right foot, left wrist, right wrist
51. pelvis, head, left wrist, right wrist
52. left foot, right foot, head, left wrist
53. left foot, right foot, head, right wrist
54. left foot, right foot, left wrist, right wrist
55. left foot, head, left wrist, right wrist
56. right foot, head, left wrist, right wrist
57. pelvis, left foot, right foot, head, left wrist
58. pelvis, left foot, right foot, head, right wrist
59. pelvis, left foot, right foot, left wrist, right wrist
60. pelvis, left foot, head, left wrist, right wrist
61. pelvis, right foot, head, left wrist, right wrist
62. left foot, right foot, head, left wrist, right wrist
63. pelvis, left foot, right foot, head, left wrist, right wrist